이동 에이전트 수행 결과 보호에 관한 연구

김희연*, 신 원†, 이경현‡
부경대학교 전자계산학과*
안철수 연구소†
부경대학교 전자컴퓨터정보통신공학부‡

A New Protection Scheme for the Results of Mobile Agent

Hee-Yeon Kim*, Weon Shin†, Kyung-Hyune Rhee‡
Dept. of Computer Science, PuKyong Nat'l University
Ahnlab,Inc.†
Division of Electronic, Computer and Telecommunication Engineering, PKNU‡
E-mail : anilove@lisia21.net, shinweon@ahnlab.com, khrhee@pknu.ac.kr

요 약

이동 에이전트 시스템은 기존의 클라이언트/서버 모델의 제한 사항들을 뛰어넘을 수 있는 대안으로 소개되었으나 보안과 같은 민감한 부분에 있어서의 보완이 미흡한 실정이다. 본 논문에서는 이동 코드로 작성되어 방문하는 호스트로부터 코드와 데이터 변경을 당할 위험이 있는 이동 에이전트의 수행 결과를 보호하기 위한 새로운 프로토콜을 제안한다.

1. 서론

인터넷과 네트워크 환경의 등장과 확산으로 전 세계는 보다 빠른 정보 공유와 작업 수행이 가능해졌으며 클라이언트/서버 모델을 기반으로 많은 발전을 해왔다. 그러나, 기존 클라이언트/서버 모델에서의 취약점[1]이 드러나면서 이를 해결하기 위한 방안으로 사용자가 처리하고자 하는 작업을 사용자를 대신하여 자율적이고 동적으로 수행하는 프로그램인 이동 에이전트가 제안되고 있으며, 이러한 이동 에이전트가 동작할 수 있는 환경을 갖춘 이동 에이전트 시스템이 등장하게 되었다. 이동 에이전트 시스템은 네트워크 부하와 트래픽, 대기시간을 줄일 수 있는 특징을 지니고 있어 분산시스템 환경에서의 응용에 적합할 것으로 기대된다. 또한 모바일 컴퓨팅, 이동 호스트 환경과 같이 지속적인 네트워크 접속이 어려운 경우에 자율적이고 비동기 적으로 수행될 수 있는 특징을 지니기 때문에 상당한 효율성을 제공할 것으로 보인다.

반면, 이러한 이동 에이전트 시스템은 기존의 클라이언트/서버 환경에 존재하는 네트워크 상에서의 메시지 수정, 호스트 보호와 같은 보안상의 문제점을 비롯하여 네트워크를 통해 원격지로 전송되어 실행되는 이동 코드(Mobile Code)로 작성되는 이동 에이전트를 도입으로써 네트워크 상과 이전해 간 호스트 상에서의 복사와 변경, 삭제의 위험 등을 추가로 안고 있다. 특히 이동 에이전트의 수행 데이터가 네트워크 상에서 악의적인 사용자에 의해 변경된 채 이동 에이전트의 소유자에게 돌아갈 경우 이동 에이전트를 파견한 사용자는 잘못된 결과를 신뢰하게 되는 심각한 문제가 발생하게 된다[2].

따라서 본 논문에서는 이동 에이전트의 수행 결과가 변경되었을 경우 이동 에이전트의 이동 경로 상의 정직한 호스트와 이동 에이전트의 소유자가 이동 에이전트의 결과에 대한 부정을 검출할 수 있는 프로토콜을 제안하고자 한다.

본 논문의 구성은 다음과 같다. 먼저 2장에서는 이동 에이전트 시스템의 취약성에 대해서 살펴보고, 3장에서는 기존에 제안된 이동 에이전트 보호 방안을 소개한 후 4장에서 이동 에이전트의 수행결과를 보호하기 위한 방안을 제시하고, 5장에서 결론을 맺는다.

2. 이동 에이전트 시스템의 취약성

본 절에서는 지금까지 알려진 이동 에이전트 시스

템의 취약성에 대하여 살펴본다. 현재 인터넷과 같은 대부분의 공개 네트워크는 안전하지 않은 것으로 알려져 있고 이동 에이전트가 수행되는 호스트 또한 반드시 정직하다는 것을 보장할 수 없는 실정이다. 그로 인해, 현재 이동 에이전트 시스템에서는 다음과 같은 보안 문제가 발생할 수 있다[3][4][5].

- **실행 호스트에 의한 이동 에이전트의 데이터 실행코드 노출 및 변경**

  에이전트가 실행되는 호스트는 에이전트의 중요한 데이터와 실행코드, 제어 흐름, 실행 결과 등을 관찰한 후 변경 또는 임의 삭제가 가능하고, 정당한 수행을 하고자 하는 이동 에이전트에게 호스트의 시스템 자원, 서비스로의 접근을 허용하지 않는 DoS 공격이 가능하다. 또한 이동 에이전트의 실행을 지연시키거나 잘못된 정보를 제공하는 공격 역시 가능하다.

- **이동 에이전트에 의한 실행 호스트의 중요 정보 노출 및 DoS 공격**

  에이전트는 정당한 접근 권한이 부여되지 않은 실행 호스트의 데이터, 메모리, 디스크 등에 접근하여 실행 호스트의 중요 정보를 삭제, 파괴, 변경할 수 있다. 또한, 실행 호스트의 자원을 독점함으로써 DoS 공격을 수행할 수 있으며 위장 채널을 사용하여 신뢰할 수 있는 에이전트로 가장하여 실행 호스트의 정보를 노출할 수 있다.

이질적인 환경을 돌아다니면서 수행되는 이동 에이전트의 특징으로 인해 이동 에이전트 시스템에서의 보안은 단순히 네트워크 상이나 에이전트 실행 호스트 상에서의 보안 뿐 아니라 이동 에이전트 자체를 보호하기 위한 방안이 상호 유기적으로 결합되어 수행되어야 한다.

3. 이동 에이전트 수행 결과 보호를 위한 기존 연구

외부의 간섭으로 인해 이동 에이전트의 데이터가 변경되거나 실행 코드, 상태 정보가 유출될 경우 정상적이고 정확한 작업 수행을 기대하기 어렵다. 따라서 이동 에이전트 실행에 대한 보호 방안이 제공되어야 하는데 아직 완전한 연구가 이루어지지 않은 실정이다. 현재까지 이동 에이전트를 보호하기 위해 연구된 방안 중 이동 에이전트 실행 후 변경을 탐지하기 위한 목적으로 제안된 방식에는 서명함수(signature function)를 이동 에이전트의 실행 가능한 코드의 부분으로 제공하는 P.Kotzanikolaou, M.Burmester, V.Chrissikopoulos의 탐지할 수 없는 서명(undetachable signature) 방안[6], 해쉬 함수의 안전성에 기반을 둔 Karjoth의 해쉬 체인을 이용한 방안[7], 해쉬 체인을 이용하여 부정 검출 및 수행 결과를 보호하기 위하여 G.Vigna의 Cryptographic Traces 방안[8]을 확장한 신원의 암호학적 추적 방안[9] 등이 있다.

P.Kotzanikolaou, M.Burmester, V.Chrissikopoulos의 방안[6]에서 이동 에이전트의 소유자는 자신의 메시지에 암호학적 해쉬 함수를 적용하여 해쉬 값을 계산하고 이 값을 이용하여 RSA 서명 값을 계산한다. 이때 계산된 값은 이동 에이전트의 데이터 부분으로 포함된다. 이 외에 RSA 서명 값을 생성하기 위한 함수와 탐지할 수 없는 서명(undetachable signature)을 생성하기 위한 함수를 추가로 생성하게 되는데 이때 탐지할 수 없는 서명 함수(undetachable signature function)는 두 개의 함수로 구성되어 있지만 분리할 수 없는 특징을 가진 준동형(homomorphic) 함수이다. 이 함수는 이동 에이전트의 실행 가능한 코드 부분으로 포함되며 이동 에이전트의 소유자는 위에서 생성한 데이터 부분과 실행 가능한 코드 부분을 네트워크를 통해 원격 호스트로 전송하게 된다. 원격 호스트는 데이터 부분에 포함된 RSA 서명 값을 검증함으로써 이동 에이전트 소유자의 신원을 확인할 수 있으며 실행 가능한 코드 부분에 포함된 두 가지 함수에 자신의 입력 값 x를 넣어 새로운 해쉬 값과 RSA 서명 값을 계산하게 된다. 이 방안은 이동 에이전트 소유자에 대한 부인 방지와 무결성은 제공하지만 상호 인증과 비밀성이 제공하지 않는 특징을 지니고 있다.

Karjoth[7]의 방안은 이동 에이전트가 방문하는 호스트로부터 얻은 각각의 수행 결과에 해쉬 함수를 적용하여 이 각각의 해쉬 함수 값이 해쉬 체인을 형성하도록 한다. 만약 이동 에이전트가 네트워크 상을 이동하던 중 악의적인 호스트로부터 공격을 받아 해쉬 체인에 변경이 가해졌을 경우 이동 에이전트의 소유자는 이 해쉬 체인을 수행하여 어떤 호스트에 의해 변경이 일어났는지 확인할 수 있다.

신원[8]의 기존의 Cryptographic Trace를 확장한 암호학적 추적 방안에서 다음 호스트로 전송되는 값은 현재 호스트의 비밀키로 현재 호스트의 ID, 목적지 호스트의 ID, 토큰을 서명한 값과 임의의 키로 이동 에이전트의 코드와 초기 상태를 암호화한 값이다. 목적지 호스트는 이 두 값을 검증한 후 이전 호스트

로부터 받은 내용과 동일한 형식의 데이터를 생성하고 자신이 받은 모든 값을 암호학적 해쉬 함수를 이용하여 해쉬 값을 계산해 낸다. 따라서 다음 호스트로 전송되는 값은 지금까지 수행되어 받은 값에 대한 해쉬 값과 현재 호스트에서 생성된 값이 된다. 이러한 과정을 거쳐 이동 에이전트의 수행 값은 사슬 관계를 형성하게 되며 한 번 수행 후에는 자신이 작성한 값이라 할지라도 수정할 수 없도록 하는 특징과 코드 삽입 또는 삭제 공격을 막을 수 있는 특징을 가지게 된다.

4. 이동 에이전트 수행 결과 보호 방안

본 논문에서는 네트워크 상의 여러 호스트를 방문하면서 수행 결과를 수집한 이동 에이전트의 수행 결과를 보호하기 위한 방안을 제안하고자 한다. 이동 에이전트의 코드와 데이터는 이동코드(Mobile Code)로 이루어진 일종의 프로그램이기 때문에 악의적인 목적을 가진 호스트이든 정직한 호스트이든 간에 이동 에이전트의 내용을 보고 변경하는 것이 가능하다.

따라서, 본 논문에서는 이동 에이전트의 내용을 보는 것은 허용하지만 악의적인 호스트가 이동 에이전트의 내용을 수정하였을 경우 이 후에 이 이동 에이전트를 수행하는 호스트와 이동 에이전트의 소유자가 이러한 변경을 검증할 수 있는 방안을 제시하고자 한다.

4.1. 수행 결과보호 프로토콜

O	이동 에이전트 소유자(Originator)의 ID
S_i	i 번째 호스트의 ID
h_i	i 번째 호스트의 결과에 대한 해쉬 값
s_i	i 번째 호스트까지의 해쉬 값을 더한 O의 서명 값
d_i	i 번째 호스트의 비밀키
n_i	i 번째 호스트의 공개 값- $p_i \times q_i$

(표 1) Notation

본 논문에서 O와 이동 에이전트가 경유하는 각각의 호스트는 서명을 생성하기 위해 소인수 분해의 어려움에 기반을 둔 RSA 디지털 서명 방식을 이용하고 이동 에이전트가 n개의 호스트를 거쳐 O에게 돌아오는 것을 가정한다. 프로토콜에서 사용하는 표기법(Notation)은 (표 1)과 같다.

(1) 서명하고자 하는 메시지 $(O, Resquest)$에 암호학적 해쉬 함수를 적용하여 얻은 h_0에 대하여 O의 RSA 개인키 d_o를 기반으로 하여 디지털 서명 $s_0 = h_0^{d_o} \bmod n_O$를 생성한다. O는 이동 에이전트가 이전할 첫 번째 호스트 S_1에게 메시지와 서명 값을 전송한다.

계산 과정
$O: \quad h_0 = hash(O, Request)$ $s_0 = h_0^{d_o} \bmod n_O$
⬇
전송 값
$O \rightarrow S_1: \quad ((O, Request), s_0)$

(표 2) Originator -> S_1의 전송 값

(2) 이전 호스트인 O로부터 메시지와 서명 값을 전송받은 호스트 S_1은 O의 공개키를 이용하여 RSA 디지털 서명을 검증하여 해쉬 값 h_0를 얻어낸다. 호스트 S_1은 해쉬 값을 복호하여 수신 받은 메시지 $(O, Resquest)$와 복호한 해쉬 값이 동일한 지 검증한다. 검증을 끝낸 후 호스트 S_1은 이동 에이전트의 요청에 대한 처리를 수행 한 후 응답($Response_1$)을 생성하게 된다. 호스트 S_1은 O가 계산했던 방식과 동일하게 메시지 $(S_1, O, Response_1)$에 해쉬 함수를 적용하여 해쉬 값 h_1을 생성하고 호스트 S_1의 서명 값 m_1을 생성한다. 호스트 S_1은 이전 호스트인 O로부터 전송 받은 값과 자신이 생성한 값을 다음 호스트인 S_2에게 전송한다.

계산 과정
$S_1: \quad h_1 = hash(S_1, O, Response_1)$ $m_1 = h_1^{d_{S_1}} \bmod n_{S_1}$ $s_1 = s_0^{h_1} \bmod n_O$
⬇
전송 값
$S_1 \rightarrow S_2: \quad ((O, Request), s_0)$ $((S_1, O, Response_1), m_1, s_1)$

(표 3) S_1 -> S_2의 전송 값

(3) 자신의 이전 호스트 S_{i-1}로 부터 메시지와 서명 값을 전송 받은 호스트 S_i는 서명을 검증하고 이동 에이전트로부터 요청에 대한 응답($Response_i$)를 생성하여 다음 호스트인 S_{i+1}에게 전송한다. 이 과정을 일반화하면 다음과 같다.

계산 과정	
S_i :	$h_i = hash(S_i, O, Response_i)$
	$m_i = h_i^{d_{S_i}} \bmod n_{S_i}$
	$s_i = s_{i-1} \times s_0^{h_i} \bmod n_O$
⬇	
전송 값	
$S_i \rightarrow S_{i+1}$:	$((O, Request), s_0)$
	$((S_1, O, Response_1), m_1, s_1)$
	⋮
	$((S_i, O, Response_i), m_i, s_i)$

(표 4) S_i -> S_{i+1}의 전송 값

(4) 호스트 S_{i+1}은 지금까지의 과정과 동일한 방식으로 서명을 생성하고 다음 호스트로 값을 전송하게 되는데 n개의 호스트를 돌아다닌 후 이동 에이전트가 O로 전송하는 값은 다음과 같다.

전송 값	
$S_n \rightarrow O$:	$((O, Request), s_0)$
	$((S_1, O, Response_1), m_1, s_1)$
	⋮
	$((S_n, O, Response_n), m_n, s_n)$

(표 5) S_n -> Originator의 전송 값

4.2. 제안 프로토콜 분석

본 절에서는 논문에서 제안한 이동 에이전트의 수행 결과를 보호하기 위한 프로토콜을 전방 무결성(Forward Integrity)의 제공, 공격자로부터의 변경 사실에 대한 검출 유무 측면에서 분석한다. 제안한 프로토콜은 다음과 같은 안전성을 가진다.

첫째, 전방 무결성(Forward Integrity)를 제공한다.

예를 들어, 공격자가 호스트 S_3의 서명 값 s_3의 비밀 키 d_{S_3}를 알고 있다고 가정했을 때, 공격자는 S_3의 서명 값을 완전히 계산해낼 수 있다. 따라서, S_3의 해쉬 값 h_3를 변경한 후 마치 자신이 올바른 호스트 S_3인 것으로 가장하여 서명 값을 생성하게 될 것이고 이후로는 호스트 S_3에 관한 한 잘못된 응답(*Response*) 값이 전달되게 될 것이다. 그러나, 공격자는 S_3 이전의 호스트 S_2, S_1의 응답 (*Response*) 값을 변경하는 것은 불가능하다. 따라서, 전방 무결성(Forward Integrity)이 제공된다.

둘째, 정직한 호스트는 이전 결과에 대한 부정을 검출할 수 있다.

정직한 호스트는 자신에게 넘어오는 값들 중 이전 호스트의 RSA 서명 값을 검증함으로써 이전 호스트가 올바른 서명을 생성했는지 검증하는 것이 가능하다.

셋째, 에이전트 소유자는 모든 결과에 대한 부정을 검출할 수 있다.

에이전트 소유자는 이동 에이전트가 n개의 호스트를 경유하고 최종적으로 전달받은 s_n 값을 계산하면 (표 6)과 같다. (표 6)에서 보는 바와 같이 s_n 값은 최초에 에이전트 소유자가 생성한 해쉬 값 h_0에 대해서 지금까지 이동 에이전트가 방문했던 호스트로부터 생성된 해쉬 값들을 합한 후 에이전트 소유자의 비밀 키 d_o로 서명한 형태이기 때문에 에이전트 소유자는 이전 호스트의 부정을 검출하는 것이 가능하다.

s_n값의 계산
$s_n = s_{n-1} \times s_0^{h_n} \bmod n_O$
$= s_{n-2} \times s_0^{h_{n-1}} \times s_0^{h_n} \bmod n_O$
$= s_0^{h_1} \times \cdots \times s_0^{h_{n-1}} \times s_0^{h_n} \bmod n_O$
$= (s_0)^{h_1 + \cdots + h_n} \bmod n_O$
$= (h_0^{d_O})^{h_1 + \cdots + h_n} \bmod n_O$
$= (h_0^{h_1 + \cdots + h_n})^{d_O} \bmod n_O$

(표 6) s_n값의 계산

5. 결론 및 향후 연구

네트워크 환경은 우리 생활의 일부분이 되었다고 할 정도로 많이 보급되었다. 많은 일들을 인터넷을 통해서 처리하게 되면서 이동 에이전트 시스템 환경은 여러 가지 이점을 가지는 편리한 도구로 소개되었다.

그러나, 이동 에이전트 시스템이 가지고 있는 여러 가지 보안 문제[4]로 인해 일반적으로 사용되기 위해서는 시스템에 대한 보안문제가 우선적으로 해결되는 것이 중요하다.

본 논문에서는 이동 에이전트 시스템이 가진 여러 가지 보안 문제 중 이동 에이전트가 수행한 결과를 보호하기 위한 프로토콜을 제안하였고, 이 프로토콜은 정직한 호스트와 이동 에이전트의 소유자가 이동 에이전트의 수행 결과의 변경 유무를 판단할 수 있도록

설계되었다. 또한 전방 무결성(Forward Integrity)를 제공하여 공격자에 의해 공격받은 부분 이외에는 무결성을 제공하도록 설계되었다. 그러나, 이동 에이전트의 수행 결과 값 자체는 평문으로 전송되기 때문에 향후 수행 결과 값을 보호하기 위한 방안이 더 연구되어야 할 것이다.

[참고문헌]

[1] 김영대, "자바로 구현하는 이동 에이전트 시스템," 프로그램 세계, 2000

[2] Wayne A. Jansen, "Countermeasures for Mobile Agent Security," Elsevier Science, 2000

[3] 신 원, "안전한 이동 에이전트 시스템의 설계와 응용," 2001

[4] Michael S. Greenverg, jennifer C. Byington, "Mobile Agents and Security," IEEE Communications Magazine, 1998

[5] 송영기, 인소란, 김명준 "소프트웨어 에이전트의 보호기술," 전자통신동향분석

[6] Panayiotis Kotzanikolaou, Mike Burmester, Vassilios Chrissikopoulos " Secure Transactions with Mobile Agents in Hostile Environments" ACISP 2000: 289-297

[7] G.Karjoth, N.Asokan, C.Gulcu, "Protecting the Computation Results of Free-roaming Agents," Proceedings of the Second International Workshop, MA'98. pp. 195-207, 1998

[8] G.Vigna, "Cryptographic Traces for Mobile Agents", In:G.Vigna (Ed.). Mobile Agents and Security, Springer-Verlag, Lecture Notes in Computer Science 1419, pp.137-153, 1998.

[9] 신 원, 이경현, "이동 에이전트 실행 보호를 위한 암호학적 추적 방안," 통신정보보호학회논문지 제 11권 제 3호, 2001